AP
JFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application No.:	09/745,023	Examiner:	Kang, Insun
Filed:	December 20, 2000	Group/Art Unit:	2193
Inventor(s):	Ram Kudukoli, Robert Dye, Paul F. Austin, Lothar Wenzel, and Jeffrey L. Kodosky	Atty. Dkt. No:	5150-44100
Title:	SYSTEM AND METHOD FOR PROGRAMMATICALLY GENERATING A GRAPHICAL PROGRAM IN RESPONSE TO PROGRAM INFORMATION		

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Commissioner for Patents, Alexandria, VA 22313-1450, on the date indicated below.

Jeffrey C. Hood

Signature 1/13/2006 Date

APPEAL BRIEF

Box: Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir/Madam:

Further to the Notice of Appeal filed October 17, 2005, and the Notification of Non-Compliant Appeal Brief of December 27, 2005, Appellant presents this Appeal Brief. Appellant respectfully requests that this appeal be considered by the Board of Patent Appeals and Interferences.

I. REAL PARTY IN INTEREST

The subject application is owned by National Instruments Corporation, a corporation organized and existing under and by virtue of the laws of the State of Delaware, and having its principal place of business at 11500 N. MoPac Expressway, Bldg. B, Austin, Texas 78759-3504.

II. RELATED APPEALS AND INTERFERENCES

No related appeals or interferences are known which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal.

III. STATUS OF CLAIMS

Claims 1 – 64 were originally filed in the application. In an amendment filed March 22, 2004, claims 1, 44, 49, 53, 56, and 60 were amended. In an amendment filed September 1, 2004, claims 1, 4, 49, 53, 56, 59, 60, and 63 were amended, and claims 3, 55, 58, and 62 were cancelled. In an amendment filed November 3, 2004, claims 1-2, 4-54, 56-57, 59-61, and 63-64 were cancelled, and new claims 65-124 were added. In an amendment filed May 16, 2005, claims 66-88, 90, 92-95, 103, 105, 111, 113, and 119 were amended, and new claims 125-145 were added. Thus, claims 65-145 are pending in the case.

Claims 65-145 stand rejected under 35 U.S.C. § 102(e) and are the subject of this appeal. A copy of claims 1-145 incorporating entered amendments, including claims 65-145 as on appeal, is included in the Claims Appendix hereto.

IV. STATUS OF AMENDMENTS

No amendments to the claims have been filed subsequent to the rejection in the Office Action of August 25, 2005. The Claims Appendix hereto reflects the current state of the claims.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The present application relates generally to the field graphical programming, and more particularly to a system and method for programmatically generating a graphical program in response to program information specifying functionality of the graphical program.

Independent claim 65 is directed to a computer-implemented method for automatically generating a new graphical program. A graphical program generation (GPG) program is executed, i.e., a program is executed that is operable to automatically generate a graphical program. *See, e.g., Specification, p. 28:10-11; Figure 4:206.* The GPG program receives information specifying functionality of the new graphical program, where the information does not specify specific objects for the new graphical program. For example, the information may specify the functionality without specifying or selecting graphical program nodes for inclusion in the new graphical program. *See, e.g., Specification, p. 28:12-14, p. 27:8-13, p. 31:1 – p. 38:17; Figure 4:208, Figure 5.* The GPG program automatically generates the new graphical program implementing the specified functionality in response to the information specifying the functionality of the new graphical program. In other words, the GPG program automatically generates the new graphical program implementing the specified functionality based on the received information. *See, e.g., Specification, p. 28:15-19, p. 31:1 – p. 38:17; Figure 4:210, Figure 5.* The new graphical program includes a plurality of interconnected nodes that visually indicate the functionality of the new graphical program. In other words, the new graphical program graphically represents the operation and functionality of the program. *See, e.g., Specification, p. 29:4-6; Figure 22.* The new graphical program is automatically generated without direct user input specifying the new graphical program. Said another way, the new graphical program is generated programmatically/automatically, i.e., by software, as opposed to manually, i.e., via user input. *See, e.g., Specification, p. 27:1-13, p. 28:15-21; Figure 4:210.*

Independent claim 112 is directed to a computer-implemented method for automatically modifying an existing graphical program. A graphical program generation (GPG) program is executed, i.e., a program is executed that is operable to automatically modify a graphical program that already exists. *See, e.g., Specification, p. 28:10-11; Figure 4:206.* During program execution, the GPG program receives information specifying functionality to add to the existing graphical program, where the information does not specify specific objects to add to the existing graphical program. For example, the information may specify the desired additional functionality without actually specifying or selecting additional graphical program nodes (or connections between nodes) for inclusion in the new graphical program. *See, e.g., Specification, p. 28:12-14, p. 27:8-13, p. 30:5-15, p. 31:1 – p. 38:17; Figure 4:208, Figure 5.* The GPG program automatically modifies the graphical program in order to implement the specified functionality, in response to receiving the information. For example, the GPG program may automatically add one or more graphical program nodes to the program, and may also automatically connect the added graphical program nodes to other graphical program nodes in the program, and possibly to each other. *See, e.g., Specification, p. 28:15-19, p. 31:1 – p. 38:17, p. 30:5-15; Figure 4:210, Figure 5.* The new graphical program includes a plurality of interconnected nodes that visually indicate the specified functionality of the existing graphical program; e.g., the new graphical program may graphically represent the operation and functionality of the program. *See, e.g., Specification, p. 29:4-6; Figure 22.* The existing graphical program is automatically modified without direct user input specifying the modification to the existing graphical program during said modifying. In other words, the new graphical program is modified programmatically/automatically, i.e., by software, as opposed to manually, i.e., via user input *See, e.g., Specification, p. 27:10-13, p. 28:19-21, p. 30:5-15.*

Independent claim 116 is directed to a computer-implemented method for automatically generating a graphical program. Information specifying functionality of the new graphical program is provided, where the information does not specify specific objects for the new graphical program. For example, the information specifies the functionality without specifying or selecting graphical program nodes for inclusion in the

new graphical program. *See, e.g., Specification, p. 28:12-14, p. 27:8-13, p. 31:1 – p. 38:17; Figure 4:208, Figure 5.* A graphical program generation (GPG) program is executed, i.e., a program is executed that is operable to automatically generate a graphical program. *See, e.g., Specification, p. 28:10-11; Figure 4:206.* Using the information, the GPG program automatically generates the new graphical program implementing the specified functionality in response to the provided information. In other words, the GPG program automatically generates the new graphical program implementing the specified functionality based on the received information. *See, e.g., Specification, p. 28:15-19, p. 31:1 – p. 38:17; Figure 4:210, Figure 5.* The new graphical program includes a plurality of interconnected nodes that visually indicate the functionality of the new graphical program. In other words, the new graphical program graphically represents the operation and functionality of the program. *See, e.g., Specification, p. 29:4-6; Figure 22.* The new graphical program is automatically generated without direct user input specifying the new graphical program. In other words, the new graphical program is generated programmatically/automatically, i.e., by software, as opposed to manually, i.e., via user input. *See, e.g., Specification, p. 27:1-13, p. 28:15-21; Figure 4:210.*

Independent claim 118 is directed to a memory medium configured with program instructions for automatically generating a new graphical program. In other words, the program instructions are executable by a processor to automatically generate a new graphical program. *See, e.g., Specification, p. 24:1-25.* The program instructions are executable to receive information specifying functionality of the new graphical program, where the information does not specify specific objects for the new graphical program. For example, the information specifies the functionality without specifying or selecting graphical program nodes for inclusion in the new graphical program. *See, e.g., Specification, p. 28:12-14, p. 27:8-13, p. 31:1 – p. 38:17; Figure 4:208, Figure 5.* The program instructions are further executable to automatically generate the new graphical program in response to the received information, where the new graphical program implements the specified functionality. In other words, the program instructions are executable by a processor to automatically generate the new graphical program implementing the specified functionality based on the received information. *See, e.g.,*

Specification, p. 28:15-19, p. 31:1 – p. 38:17; Figure 4:210, Figure 5. The new graphical program includes a plurality of interconnected nodes that visually indicate the functionality of the new graphical program. In other words, the new graphical program graphically represents the operation and functionality of the program. *See, e.g., Specification, p. 29:4-6; Figure 22.* The new graphical program is automatically created without direct user input specifying the new graphical program. said another way, the new graphical program is generated programmatically/automatically, i.e., by software, as opposed to manually, i.e., via user input. *See, e.g., Specification, p. 27:10-13, p. 28:15-21; Figure 4:210.*

Independent claim 121 is a system analogue claim to claim 65, and is directed to a system for automatically generating a new graphical program. A processor is coupled to a memory that stores a graphical program generation (GPG) program. In other words, the memory stores a program that is executable by the processor to automatically generate a new graphical program. *See, e.g., Specification, p. 19:6-9, p. 24:1-25, p. 26:3-9; Figure 3.* The processor is operable to execute the GPG program to receive information specifying functionality of the new graphical program, where the information does not specify specific objects for the new graphical program. For example, the information specifies the functionality without specifying or selecting graphical program nodes for inclusion in the new graphical program. *See, e.g., Specification, p. 28:12-14, p. 27:8-13, p. 31:1 – p. 38:17; Figure 4:208, Figure 5.* The program is further executable to automatically generate the new graphical program in response to the received information, where the new graphical program implements the specified functionality. In other words, the GPG program is executable by the processor to automatically generate the new graphical program implementing the specified functionality based on the received information. *See, e.g., Specification, p. 28:15-19, p. 31:1 – p. 38:17; Figure 4:210, Figure 5.* The new graphical program includes a plurality of interconnected nodes that visually indicate the functionality of the new graphical program. In other words, the new graphical program graphically represents the operation and functionality of the program. *See, e.g., Specification, p. 29:4-6; Figure 22.* The new graphical program is automatically created without direct user input specifying the new graphical program.

said another way, the new graphical program is generated programmatically/automatically, i.e., by software, as opposed to manually, i.e., via user input. *See, e.g., Specification, p. 27:10-13, p. 28:19-21; Figure 4:210.*

Independent claim 135 is directed to a computer-implemented method for automatically generating a new graphical program. In other words, the method for automatically generating a new graphical program is performed by a computer, e.g., via execution of program instructions. *See, e.g., Specification, p. 19:6-9, p. 24:1-25; Figures 1 and 3.* Information specifying functionality of the new graphical program is received, where the information does not specify specific objects for the new graphical program. For example, the information may specify the functionality without specifying or selecting graphical program nodes for inclusion in the new graphical program. *See, e.g., Specification, p. 27:8-13, p. 28:12-14, p. 31:1 – p. 38:17; Figure 4:208, Figure 5.* The new graphical program implementing the specified functionality is generated in response to the information specifying the functionality of the new graphical program. In other words, the method automatically generates the new graphical program implementing the specified functionality based on the received information. *See, e.g., Specification, p. 28:15-19, p. 31:1 – p. 38:17; Figure 4:210, Figure 5.* The new graphical program includes a plurality of interconnected nodes that visually indicate the functionality of the new graphical program. In other words, the new graphical program graphically represents the operation and functionality of the program. *See, e.g., Specification, p. 29:4-6; Figure 22.* The new graphical program is automatically generated without direct user input including the plurality of nodes or connecting the plurality of nodes. Said another way, the new graphical program is generated programmatically/automatically, i.e., by the computer executing software, as opposed to manually, i.e., via user input. *See, e.g., Specification, p. 27:10-13, p. 28:15-21; Figure 4:210.*

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 65-145 were rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,366,300 to Ohara et al ("Ohara").

VII. ARGUMENT

Section 103(a) Rejection

Claims 65-145 were rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,366,300 to Ohara et al ("Ohara"). Appellant respectfully traverses this rejection for the following reasons. Different groups of claims are addressed under their respective subheadings.

Claims 65, 72, 73, 78, 83, 92, 116, 121, 129, 131

Claim 65 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

The Examiner asserts that Ohara discloses all the features and limitations of claim 65. Appellant respectfully submits that there are numerous limitations of claim 65 not taught or suggested by Ohara.

For example, the Examiner asserts that Ohara teaches "the GPG program receiving information, wherein the information specifies functionality of the new graphical program", citing Figure 4 and Figure 6, but omits the claimed feature "wherein the information does not specify specific objects for the new graphical program".

Appellant submits that while Figures 4 and 6 disclose user input specifying functionality of the program, these figures (and Ohara in general) fail to teach "wherein the information does not specify specific objects for the new graphical program" as claimed. Figure 4 is described thusly (col. 16:66 – col. 17:10):

> FIG. 4 is an ***explanatory diagram showing a behavioral characteristic selection window 300 for selecting a behavior of the output signal Y000 selected from the signal box 200***. The behavioral characteristic selection window 300 shown in FIG. 4 is displayed on the display apparatus 1 when a signal set item of a set, menu on the signal box 200 shown in FIG. 3 is selected. It should be noted that the signal set item itself is not shown in FIG. 3.
>
> ***The user selects and enters a behavioral characteristic 301, a behavioral state 302 and a comment 303 of the object defining a behavior, which has been selected from the signal box 200*** shown in FIG. 3, as follows. (*emphasis added*)

Figure 6 is described thusly:

> FIG. 6 is an explanatory diagram showing a relevant signal selection window 500, a window used for selecting signals relevant to the behavior of the output signal. ***A signal relevant to the behavior of the output signal is selected from those shown in the signal box 200 described earlier by clicking a graphical object representing the signal by means of the mouse*** 21. ***A graphical object representing a selected signal is automatically displayed in a behavior relevant signal area*** 501 of the relevant signal selection window 500 as shown in FIG. 6. In the case of some contemporary visual programming tools, in such an operation to select an item signal, it is necessary to carry out an operation of dragging a graphical object representing the selected item to a predetermined position typically by means of a mouse. In the case of the visual programming method provided by the present invention and the system adopting the method, on the other hand; the dragging operation is not required.
>
> In the explanatory diagram of FIG. 6, ***input signals X000, X001 and X002 are selected by repeating the select operation described above.*** The input signals X000, X001 and X002 are signals generated by on/off switches for operating the output signal Y000. The switches correspond to switches of a generally used illuminator.
> (*emphasis added*)

Nowhere do Figures 4 and 6, nor the related text (or Ohara in general) disclose these features of claim 65. Rather, the Figures and cited text emphasize the fact that direct user input selects virtually all aspects of the generated program, as is also clearly described elsewhere.

For example, Figure 2 and related text describe the basic method of Ohara, and disclose user input specifying objects for the program:

> With the visual programming method provided by the present invention, in order to automatically generate a program of a PLC, ***the user executes the steps of selecting an output signal defining a behavior; selecting a behavior; selecting signals relevant to the behavior; setting behavioral rules; and confirming the behavior.*** For operations carried out to execute the steps, the use of various input units can be thought of. Graphical objects such as a button, a check box, a slide bar and an icon are operated mainly by clicking or dragging the mouse 21 of the input apparatus 2. On the other hand, a string of characters is entered mainly by using the keyboard 22 of the input apparatus 2. (*emphasis added*)

More particularly disclosing such user selection, col. 6:19-23 reads:

Further, a system adopting the visual programming method according to the present invention comprises:

an object selection means which displays graphical objects each defining a behavior on a graphical editor and which allows ***a user to select those of the displayed graphical objects as objects to be used in the programming work***;

a behavior selection means which displays behavioral characteristics of the graphical objects and which allows ***the user to select a behavioral characteristic of each of the selected graphical objects***;

a relevant object selection means which defines a relation among the plurality of graphical objects selected by using the object selection means and the behavior selection means; and

a behavioral rule setting means which sets a behavioral rule of the selected graphical objects.
(*emphasis added*)

Additionally, claim 1 of Ohara recites in pertinent parts:

1. A visual programming method comprising:

a first step of displaying graphical objects representing a plurality of members connected to output terminals of a programmable logic controller;

a second step of displaying graphical objects representing the output terminals of the programmable logic controller;

a third step of ***connecting a first graphical object selected by a user from the graphical objects representing the plurality of members displayed in the first step, to a second graphical object selected by the user*** from the graphical objects representing the output terminals and displayed in the second step;

a fourth step of ***allowing a user to select a graphical object or a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram;

a sixth step of creating a layout diagram showing a layout of the graphical objects representing the plurality of members and displaying said layout diagram on a screen;

a seventh step of ***allowing the user to select one of said graphical objects*** each used for defining a function and transferring said selected graphical object to said layout diagram created and displayed at said sixth step;

an eighth step of ***allowing a user to select a plurality of graphical objects from said graphical objects*** each used for defining a function and

displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram created at said sixth step;

a ninth step of ***allowing a user to select a graphical object or a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram created at said sixth step as well as displaying the same plurality of selected graphical objects on said layout diagram;

a tenth step of identifying arrangement orders of ***graphical objects selected by a user*** and assigning a priority to each of said arrangement orders when a plurality of arrangement orders are identified;

an eleventh step of displaying the same plurality of identified arrangement orders sequentially one after another in a sequence determined by said priorities assigned thereto at said tenth step,

a twelfth step of ***allowing the user to select an arrangement order among the same plurality of identified arrangement orders*** displayed sequentially at said eleventh step;

a thirteenth of ***allowing a user to change an arrangement order of graphical objects selected by the user*** at said twelfth step;

a fourteenth step of detecting a settable parameter graphical object with a new parameter thereof ***among graphical objects selected by a user*** and notifying the user of a result of detection;

a fifteenth step of detecting selection of said graphical object reported to the user at said fourteenth step or a graphical object other than said reported graphical object and creating and displaying a window used for setting a parameter of said reported graphical object or said other graphical object; and

a sixteenth step of ***allowing the user to set said parameter*** on said window displayed at said fifteenth step.

Thus, Appellant submits that Ohara does not teach this feature of claim 65.

The Examiner further asserts that Ohara teaches "wherein said automatically generating the new graphical program is performed without direct user input specifying the new graphical program", citing col. 16:27-28, and col. 17:26-39. However, col. 16:27-28 actually reads:

> "In the following description, only matters related to the user interface characterizing the visual programming method are explained, excluding automatic generation of a program ***because the automatic generation of a program is the same as the conventional system***." (*emphasis added*)

In other words, Ohara does not consider the automatic generation of the program in Ohara's system and method to be novel, and does not disclose how it is performed, other than to say it is performed according to prior art approaches.

Col. 17:26-39 reads:

> Subsequently, the user enters a comment 303 to a character input area, a field commonly known in the visual interface environment. In the case of creation of a program implementing a quiz response system, for example, the comment can be used for giving a name of `correct answer lamp` to the output signal Y000. The user enters a string of characters as a comment 303 to the character input area via the keyboard 22. The comment 303 is not absolutely required though. That is to say, a comment 303 does not have to be entered. In the above description, the behavioral characteristic 301, the behavioral state 302 and the comment 303 are entered in an order they are enumerated here. It should be noted, however, that they can be entered in an order freely selected by the user.

Clearly, the cited text in no way discloses "wherein said automatically generating the new graphical program is performed without direct user input specifying the new graphical program", as recited in claim 65.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 65. Appellant notes that the above arguments are also applicable to the other independent claims grouped therewith.

Claims 66, 117, 122

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said automatically generating the new graphical program is performed without direct user input selecting the plurality of nodes and without direct user input specifying the interconnections between the plurality of nodes".

In asserting that Ohara teaches these features, the Examiner cites col. 16:27-28, and col. 17:26-39. However, as shown above with regard to claim 65, the cited text in no way discloses this feature. Nor, as argued at length above, does Ohara anywhere teach or suggest this feature, but rather specifically describes user-selection of objects in the generated program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 66.

Claims 67, 123

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the new graphical program comprises a block diagram portion comprising the plurality of interconnected nodes and a user interface portion; and wherein said automatically generating the new graphical program includes automatically generating the block diagram portion and the user interface portion".

In asserting that Ohara teaches these features, the Examiner cites Figure 17 and col. 3:46-56. However, Appellant notes that Figure 17 is described thusly:

> FIG. 17 is an explanatory diagram showing a portion of the component model window 160. On the component model window 160 shown in FIG. 17, typically, combinations of behavior relevant signals and behavioral characteristics displayed on the behavior verification window 1300 shown in FIG. 14 are expressed globally on a macro basis. Since the purpose of the behavior verification window 1300 is to locally verify operations carried out by a program generated for an output signal, a logic equation and behavioral characteristics are displayed in detail. On the other hand, the component model window 160 is used for verification of setting of behavioral characteristics for all output signals globally on a macro basis. Pieces of information set for behavioral characteristics of all output signals are thus displayed on the component model window 160 globally on a macro basis in order to allow all the pieces of information to be displayed at the same time. It should be noted that, in the example shown in FIG. 17, information on only one output signal is displayed though. (col. 38:65 – col. 39:16)

Appellant respectfully submits that Ohara nowhere describes Figure 17 as a graphical program including a block diagram portion comprising the plurality of interconnected nodes and a user interface portion. Rather, as described in the cited text, "the component model window 160 is used for verification of setting of behavioral characteristics for all output signals globally on a macro basis", where "Pieces of information set for behavioral characteristics of all output signals are thus displayed on the component model window 160 globally on a macro basis in order to allow all the pieces of information to be displayed at the same time." Appellant thus respectfully submits that the Examiner's characterization of Figure 17 as a graphical program with a

block diagram and user interface portion is incorrect, and is not supported by the cited text.

Col. 3:46-56 reads:

> A function block diagram (FBD) is a program created by a language whereby function blocks (FBs), graphical objects each representing a function, are connected to each other to form a flow of data processing. A graphical editor for this language is a tool providing a work environment in which programming can be carried out by operating graphical objects displayed on a screen. The graphical objects include icons and characters used in the graphical user interface (GUI), a commonly known user interface for personal computers and workstations.

This text describes prior art function block diagrams that may be created by a user via a graphical editor, and in no way discloses an automatically generated graphical program that includes a block diagram and a user interface portion. The only user interface mentioned in this text is that of the work environment, i.e., of the programming environment.

Applicant respectfully submits that the Examiner has generally mischaracterized Ohara. For example, Applicant notes that the user interfaces described by Ohara here and elsewhere are not part of the generated program, but rather are comprised in various portions of Ohara's development system or tools used to generate and/or verify the generated program, e.g., Ohara's "behavior verification unit", mentioned in the cited portion of Ohara, Ohara's . As Ohara describes with reference to Figure 2 in col. 20, lines 50-60:

> At the step ST6, results of the operations carried out at the steps ST1 to ST5, that is, results of the programming work, are verified. When the setting of behavioral rules is finished, a behavior verification window 1300 shown in FIG. 14 is displayed on the display apparatus 1. On the behavior verification window 1300, the 3 input signals X000, X001 and X002, that is, the behavior relevant signals of the output signal Y000, are displayed as a rudder diagram. That is to say, the behavior verification window 1300 which displays behavioral characteristics of the output signal appears on the display apparatus 1.

Applicant notes that step ST6 of Figure 2 is simply a verification step in the development process of Ohara, and respectfully submits that the user interface of the behavior verification unit is *not* a user interface portion of Ohara's generated program. Applicant notes that Ohara's Figure 66 illustrates "a typical user interface of the behavior verification unit" (col. 61, lines 9-10). Applicant submits that Ohara's behavior verification unit is specifically *not* a portion of a generated program, but rather is a tool used to verify the functionality of the generated program, and notes that other tools for aiding in the generation of the program described in Ohara include: a "behavior characteristic selection unit", "behavioral object selection unit", "behavioral condition setting unit", and a "relevant object selection unit". Applicant further notes that one or more of these additional tools also includes a respective graphical user interface which is similarly *not* a portion of the generated program. Applicant can find no reference teaching or suggesting that the generated program includes a graphical user interface portion, nor any reference teaching or suggesting that automatically generating the program includes generating a graphical user interface portion.

Additionally, Appellant notes that a program generated for, transferred to, and executed on, a programmable logic controller (PLC) would not include such a graphical user interface portion, such PLCs lacking display functionality.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 67.

Claims 68, 124

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said automatically generating the new graphical program comprises: creating a plurality of graphical program objects in the new graphical program; and interconnecting the plurality of graphical program objects in the new graphical program; wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites Figure 17 and col. 3:46-56. However, as argued above with regard to claim 67, nowhere does Figure 17 disclose automatically creating a plurality of graphical program objects in the new

graphical program; and interconnecting the plurality of graphical program objects in the new graphical program. Rather, as described in the cited text, "the component model window 160 is used for verification of setting of behavioral characteristics for all output signals globally on a macro basis", where "Pieces of information set for behavioral characteristics of all output signals are thus displayed on the component model window 160 globally on a macro basis in order to allow all the pieces of information to be displayed at the same time." Appellant thus respectfully submits that the Examiner's characterization of Figure 17 resulting from the *automatic creation* of a plurality of graphical program objects is incorrect, and is not supported by the cited text. Rather, as argued at length above, in Ohara's system, the *user* selects the graphical program objects.

Moreover, as also shown above, col. 3:46-56 describes prior art function block diagrams that may be created by a user via a graphical editor, and in no way discloses automatically generating a new graphical program by creating a plurality of graphical program objects in the new graphical program and interconnecting them, where the interconnected plurality of graphical program objects form at least a portion of the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 68.

Claim 69

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said automatically generating the new graphical program comprises: creating one or more user interface objects in the new graphical program, wherein the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:10, and col. 5:32-41. However, col. 16:66 – col. 17:10 actually reads:

> FIG. 4 is an ***explanatory diagram showing a behavioral characteristic selection window 300 for selecting a behavior of the output signal Y000 selected from the signal box 200***. The behavioral characteristic selection window 300 shown in FIG. 4 is displayed on the display apparatus 1 when a signal set item of a set, menu on the signal box 200 shown in FIG. 3 is

selected. It should be noted that the signal set item itself is not shown in FIG. 3.

The user selects and enters a behavioral characteristic 301, a behavioral state 302 and a comment 303 of the object defining a behavior, which has been selected from the signal box 200 shown in FIG. 3, as follows. (*emphasis added*)

Col. 5:32-41 reads:

As a result, even a user who has a lack of professional knowledge of programming is capable of developing a program with ease.

In addition, in the visual programming method according to the present invention, the step of selecting a behavioral characteristic of a graphical object displayed on the graphical editor further includes a step of entering a detailed parameter of the behavioral characteristic. As a result, ***the user is capable of changing a set value of a behavioral characteristic and, hence, creating a program*** with ease.

Moreover, in the visual method according to the present invention, the step of setting a behavioral rule for graphical objects each defining a behavior with respect to the graphical objects further includes a step of generating a behavioral rule not set yet on the graphical editor from already set behavioral rules. As a result, ***the user is capable of creating a program*** with ease without the need to take all behavioral rules into consideration. (*emphasis added*)

As the cited text (and the text and claim quoted above with regard to claim 65) clearly indicates, Ohara's system and method relies on direct user input specifying virtually all aspects of the generated program, and nowhere discloses automatic generation of a graphical in the manner claimed.

For example, nowhere does Ohara disclose that the generated program includes a user interface. As argued above with regard to claim 67, the only user interfaces disclosed by Ohara are those of the work environment, i.e., of the programming environment. In other words, Applicant respectfully submits that the Examiner has generally mischaracterized Ohara, in that the user interfaces described by Ohara are not part of the generated program, but rather are comprised in various portions of Ohara's development system or tools used to generate and/or verify the generated graphical program. Applicant can find no reference teaching or suggesting that the generated

program includes automatically generated user interface objects for providing input to and/or displaying output from the generated program.

Additionally, as noted above, a program generated for, transferred to, and executed on, a programmable logic controller (PLC) would not include such user interface objects, such PLCs lacking display functionality.

Thus, nowhere does the cited text (or Ohara in general) disclose automatically generating the new graphical program by creating one or more user interface objects in the new graphical program, where the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 69.

Claim 70

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the new graphical program is a virtual instrument".

In asserting that Ohara teaches these features, the Examiner cites col. 65:50-57, which reads:

> INDUSTRIAL APPLICABILITY
>
> As described above, the visual programming method provided by the present invention and the system adopting the method are based on a technique of user interfacing that allows a program to be generated automatically by selecting an object for defining a behavior, defining a behavior and setting a behavioral rule on a graphical editor.

Nowhere does the cited text (or Ohara in general) teach or suggest that the generated program is a virtual instrument. In fact, Ohara never mentions an instrument at all, and more specifically fails to disclose or even hint at a virtual instrument.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 70.

Claim 71

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the GPG program is a graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:10, and col. 5:32-41, quoted above with regard to claim 69.

Nowhere does the cited text (or Ohara in general) teach or suggest that the program that generates the program is a graphical program. In other words, Ohara does not disclose that the generating program (as opposed to the generated program) includes a plurality of interconnected nodes that visually represent the functionality of the program. Appellant respectfully submits that the Examiner has improperly characterized a program that utilizes graphics as a "graphical program". As is well known in the art of programming, both text-based and graphical programs may utilize graphics, e.g., graphical user interfaces, and so this is not a distinguishing characteristic of graphical programs in general.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 71.

Claim 74

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the information received by the GPG program specifies a state diagram; and wherein the GPG program is operable to generate a new graphical program that implements the specified state diagram".

In asserting that Ohara teaches these features, the Examiner cites col. 65:50-57, quoted above with regard to claim 70. However, the cited text does not teach or suggest, or even hint at, information specifying a state diagram, nor automatically generating a graphical program implementing the state diagram. In fact, Ohara never even mentions state diagrams at all.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 74.

Claim 75

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the information received by the GPG program specifies a

prototype; and wherein the GPG program is operable to generate a new graphical program that implements the specified prototype".

In asserting that Ohara teaches these features, the Examiner cites col. 65:50-57, quoted above with regard to claim 70. However, the cited text does not teach or suggest, or even hint at, information specifying a prototype, nor automatically generating a graphical program implementing the prototype. In fact, Ohara never even mentions a prototype at all.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 75.

Claim 76

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the information received by the GPG program specifies a test executive sequence; and wherein the GPG program is operable to generate a new graphical program that implements the specified test executive sequence".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claim 65. However, the cited text does not teach or suggest, or even hint at, information specifying a test executive sequence, nor automatically generating a graphical program implementing the specified test executive sequence. In fact, Ohara never even mentions a test executive sequence at all, nor any kind of test.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 76.

Claim 77

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said GPG program receiving information comprises the GPG program receiving user input specifying desired functionality of the new graphical program; and wherein the GPG program is operable to generate a new graphical program that implements the specified desired functionality".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claim 65. However, the cited text does not teach or suggest, or even hint at, receiving user input specifying desired functionality of the new graphical program, nor generating a new graphical program that implements the specified desired functionality, given that, as claimed, the information input by the user does not specify specific objects for the new graphical program, whereas in Ohara's system and method, the user specifies or selects the specific objects for the generated program, as argued at length above with regard to claim 65.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 77.

Claims 79, 80

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the information received by the GPG program specifies an instrumentation function; and wherein the GPG program is operable to generate a new graphical program that implements the specified instrumentation function".

In asserting that Ohara teaches these features, the Examiner cites col. 65:50-57, quoted above with regard to claim 70. However, the cited text does not teach or suggest, or even hint at, information specifying an instrumentation function, nor automatically generating a graphical program implementing the instrumentation function. In fact, Ohara never even mentions an instrumentation function at all.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 79.

Claim 81

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the information received by the GPG program comprises information regarding an existing program having program functionality; and wherein the GPG program is operable to generate a new graphical program that implements at least a portion of the program functionality of the existing program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claim 65. However, the cited text does not teach or suggest, or even hint at, receiving an existing program (with program functionality) specifying desired functionality of the new graphical program, nor generating a new graphical program that implements at least a portion of the program functionality of the existing program. In fact, nowhere does Ohara mention or even hint at automatically generating a new graphical program based on an existing program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 81.

Claim 82

In addition to the distinctions noted above with regard to claim 81, the cited art fails to teach "wherein the existing program is a graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 65:50-57, quoted above with regard to claim 70. However, as noted above with regard to claim 81, the cited text (and Ohara in general) does not disclose receiving an existing program specifying desired functionality of the new graphical program at all, and more specifically does not teach or suggest, or even hint at, receiving an existing *graphical* program specifying the desired functionality of the new graphical program, nor generating a new graphical program that implements at least a portion of the program functionality of the existing graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 82.

Claims 84, 85

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the new graphical program generated by the GPG program has program functionality; and wherein the GPG program is operable to determine at least a portion of the program functionality independently of the received information".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8. However, the cited text actually reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

Clearly, the cited text does not teach or suggest, or even hint at, a GPG program determining at least a portion of the program functionality of a new graphical program independently of the received information.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 84.

Claim 86

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the new graphical program comprises graphical program code; and wherein the GPG program is operable to receive code generation information specifying how to generate at least a portion of the graphical program code".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

Clearly, the cited text does not teach or suggest, or even hint at, a GPG program is operable to receive code generation information specifying how to generate at least a portion of the graphical program code.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 86.

Claim 87

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said GPG program automatically generating the new graphical program comprises the GPG program calling an application programming interface (API) enabling the automatic generation of a graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claim 65. However, the cited text does not teach or suggest, or even hint at, a program calling an application programming interface (API) enabling the automatic generation of a graphical program. In fact, nowhere does Ohara mention or even hint at an API at all.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 87.

Claim 88

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said GPG program automatically generating the new graphical program comprises the GPG program automatically requesting a server program to generate the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 25:8-10. However, the cited text actually reads:

> A generated program is transferred to a PLC typically by way of the network connection apparatus 5 shown in FIG. 1.

Clearly, the cited text does not disclose a (GPG) program automatically requesting a server program to generate the new graphical program. Rather, the cited text describes deploying a generated program to a programmable logic controller (PLC) via a network. Appellant notes that Ohara never mentions a server program at all.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 88.

Claim 89

In addition to the distinctions noted above with regard to claim 88, the cited art fails to teach "wherein the server program is an application instance of a graphical programming environment".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:65. However, the cited text (also quoted above with regard to claim 65) actually reads:

> FIG. 4 is an ***explanatory diagram showing a behavioral characteristic selection window 300 for selecting a behavior of the output signal Y000 selected from the signal box 200.*** The behavioral characteristic selection window 300 shown in FIG. 4 is displayed on the display apparatus 1 when a signal set item of a set, menu on the signal box 200 shown in FIG. 3 is selected. It should be noted that the signal set item itself is not shown in FIG. 3.
>
> ***The user selects and enters a behavioral characteristic 301, a behavioral state 302 and a comment 303 of the object defining a behavior, which has been selected from the signal box 200*** shown in FIG. 3, as follows. (*emphasis added*)

Clearly, the cited text does not disclose a server program being an application instance of a graphical programming environment, as claimed, i.e., where the server program is callable by a program (GPG program) to automatically generate a new graphical program. As Appellant noted above, Ohara never mentions a server program at all.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 89.

Claim 90

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the GPG program comprises a client portion and a server portion; and wherein the client portion is operable to utilize an application programming interface (API) in order to direct the server program to automatically generate the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:65. However, the cited text, quoted above with regard to claim 89, nowhere discloses these features. Appellant notes that Ohara never mentions or even hints at either a server or a client, and specifically fails to disclose a client portion of a program using an API to direct a server program to automatically generate the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 90.

Claim 91

In addition to the distinctions noted above with regard to claim 90, the cited art fails to teach "wherein the client portion of the GPG program executes in a first computer system; wherein the server portion of the GPG program executes in a second computer system; and wherein the first computer system is connected to the second computer system".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:65. However, the cited text, quoted above with regard to claim 89, nowhere discloses these features. As noted above, Ohara never mentions or even hints at either a server or a client, and specifically fails to disclose a client portion of a program executing in a first computer system using an API to direct a server program executing in a second computer system connected to the first computer system to automatically generate the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 91.

Claim 93

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the new graphical program implements only a portion of the specified functionality".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

The cited text does not teach or suggest, or even hint at, an automatically generated graphical program that implements only a portion of the specified functionality, as claimed.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 93.

Claim 94

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the new graphical program is a partial program, the method further comprising: adding additional graphical code to the new graphical program, in response to manual user input, in order to complete the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

The cited text does not teach or suggest, or even hint at these claimed features.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 94.

Claim 95

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said automatically generating the new graphical program comprises including at least one graphical program object in the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

The cited text does not teach or suggest automatically including at least one graphical program object in the new graphical program as claimed, i.e., in response to information that does not specify specific objects for the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 95.

Claim 96

In addition to the distinctions noted above with regard to claim 95, the cited art fails to teach "wherein the new graphical program includes a block diagram, wherein the at least one graphical program object comprises a function node placed in the block diagram".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

The cited text does not teach or suggest automatically including a function node in a block diagram as claimed, i.e., in response to information that does not specify specific objects for the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 96.

Claim 97

In addition to the distinctions noted above with regard to claim 95, the cited art fails to teach "wherein the new graphical program includes a block diagram, wherein the at least one graphical program object comprises a automatic structure placed in the block diagram".

In asserting that Ohara teaches these features, the Examiner cites col. 65:50-57, quoted above with regard to claim 70. The cited text does not teach or suggest automatically including an automatic structure in a block diagram as claimed, i.e., in response to information that does not specify specific objects for the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 97.

Claim 98

In addition to the distinctions noted above with regard to claim 95, the cited art fails to teach "wherein the new graphical program includes a user interface panel,

wherein the at least one graphical program object comprises a user interface object placed in the user interface panel".

In asserting that Ohara teaches these features, the Examiner cites col. 7:22-36, col. 7:55-59, and Figure 37.

The cited passages read:

> Further, a visual programming method according to the present invention comprises:
>
> a first step of creating and displaying graphical objects each defining a behavior on a graphical editor used for carrying out programming work by operating graphical objects each visually representing a function;
>
> a second step of creating a layout diagram showing a layout of graphical objects representing configuration members of an application system being created by a user and displaying the layout diagram on the same screen as a screen used at the first step; and
>
> a third step of allowing the user to select one of the graphical objects each used for defining a behavior and displayed at the first step and transferring the selected graphical object to the layout diagram.
> ...
>
> a sixth step of creating a layout diagram showing a layout of graphical objects representing configuration members of an application system being created by a user and displaying the layout diagram on a screen different from a screen used at the first step; and...

The cited text does not teach or suggest an automatically generated graphical program that includes a user interface panel, nor automatically placing a user interface object in the user interface panel, as claimed, i.e., in response to information that does not specify specific objects for the new graphical program, but rather, discloses user selection of graphical objects for inclusion in a layout diagram. Nowhere are these objects described as user interface objects, nor does the cited text describe the generated program as including a user interface panel.

Nor does Figure 37 disclose these features of claim 98.

Moreover, as Appellant argued above with regard to claims 67 and 69, Ohara's generated program has no user interface, and so cannot teach these features of claim 98.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 98.

Claim 99

In addition to the distinctions noted above with regard to claim 98, the cited art fails to teach "wherein the user interface object is a user interface input object placed in the user interface panel for performing one or more of: viewing input to the new graphical program; or providing input to the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 7:22-36, col. 7:55-59, and Figure 37.

However, the cited passages, quoted above with regard to claim 98, nowhere disclose an automatically generated graphical program that includes a user interface panel, nor automatically placing a user interface input object in the user interface panel for viewing input to the new graphical program or providing input to the new graphical program, as claimed, i.e., in response to information that does not specify specific objects for the new graphical program, but rather, discloses user selection of graphical objects for inclusion in a layout diagram. Nowhere are these objects described as user interface input objects, specifically, for viewing or providing input to the new graphical program.

Nor does Figure 37 disclose these features of claim 99.

Moreover, as Appellant argued above with regard to claims 67 and 69, Ohara's generated program has no user interface, and so cannot teach these features of claim 99.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 99.

Claim 100

In addition to the distinctions noted above with regard to claim 98, the cited art fails to teach "wherein the user interface object is a user interface output object placed in the user interface panel for viewing output of the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 7:22-36, col. 7:55-59, and Figure 37.

However, the cited passages, quoted above with regard to claim 98, nowhere disclose an automatically generated graphical program that includes a user interface panel, nor automatically placing a user interface output object in the user interface panel for viewing output of the new graphical program, as claimed, i.e., in response to information that does not specify specific objects for the new graphical program, but rather, discloses user selection of graphical objects for inclusion in a layout diagram. Nowhere are these objects described as user interface output objects, specifically, for viewing output of the new graphical program.

Nor does Figure 37 disclose these features of claim 100.

Moreover, as Appellant argued above with regard to claims 67 and 69, Ohara's generated program has no user interface, and so cannot teach these features of claim 100.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 100.

Claim 101

In addition to the distinctions noted above with regard to claim 98, the cited art fails to teach "wherein the new graphical program also includes a block diagram, wherein the user interface object is a user interface input object placed in the user interface panel for performing one or more of: viewing input to the block diagram; or providing input to the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 7:22-36, col. 7:55-59, and Figure 37.

However, the cited passages, quoted above with regard to claim 98, nowhere disclose an automatically generated graphical program that includes a user interface panel, nor automatically placing a user interface input object in the user interface panel for viewing input to a block diagram included in the new graphical program or providing input to the block diagram of the new graphical program, as claimed, i.e., in response to information that does not specify specific objects for the new graphical program, but rather, discloses user selection of graphical objects for inclusion in a layout diagram. Nowhere are these objects described as user interface input objects, specifically, for viewing or providing input to the block diagram of the new graphical program.

Nor does Figure 37 disclose these features of claim 101.

Moreover, as Appellant argued above with regard to claims 67 and 69, Ohara's generated program has no user interface, and so cannot teach these features of claim 101.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 101.

Claim 102

In addition to the distinctions noted above with regard to claim 98, the cited art fails to teach "wherein the new graphical program also includes a block diagram, wherein the user interface object is a user interface output object placed in the user interface panel for viewing output from the block diagram".

In asserting that Ohara teaches these features, the Examiner cites col. 7:22-36, col. 7:55-59, and Figure 37.

However, the cited passages, quoted above with regard to claim 98, nowhere disclose an automatically generated graphical program that includes a user interface panel, nor automatically placing a user interface output object in the user interface panel for viewing output from a block diagram included in the new graphical program, as claimed, i.e., in response to information that does not specify specific objects for the new graphical program, but rather, discloses user selection of graphical objects for inclusion in a layout diagram. Nowhere are these objects described as user interface output objects, specifically, for viewing output from the block diagram included in the new graphical program.

Nor does Figure 37 disclose these features of claim 102.

Moreover, as Appellant argued above with regard to claims 67 and 69, Ohara's generated program has no user interface, and so cannot teach these features of claim 102.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 102.

Claims 103, 104

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach (automatically) "including a first graphical program object and a second

graphical program object in the new graphical program; and connecting the first graphical program object to the second graphical program object".

In asserting that Ohara teaches these features, the Examiner cites col. 16:27-28, and col. 17:26-39. However, as noted above, col. 16:27-28 actually reads:

> "In the following description, only matters related to the user interface characterizing the visual programming method are explained, excluding automatic generation of a program ***because the automatic generation of a program is the same as the conventional system***." (*emphasis added*)

Thus, Ohara does not consider the automatic generation of the program in Ohara's system and method to be novel, and does not disclose how it is performed, other than to say it is performed according to prior art approaches. Nowhere does this passage disclose these features of claim 103.

Col. 17:26-39 reads:

> Subsequently, the user enters a comment 303 to a character input area, a field commonly known in the visual interface environment. In the case of creation of a program implementing a quiz response system, for example, the comment can be used for giving a name of `correct answer lamp` to the output signal Y000. The user enters a string of characters as a comment 303 to the character input area via the keyboard 22. The comment 303 is not absolutely required though. That is to say, a comment 303 does not have to be entered. In the above description, the behavioral characteristic 301, the behavioral state 302 and the comment 303 are entered in an order they are enumerated here. It should be noted, however, that they can be entered in an order freely selected by the user.

Clearly, the cited text in no way discloses automatically including graphical program objects in the new graphical program and automatically connecting them, as recited in claim 103, i.e., without information specifying the objects, and without direct user input specifying the graphical program.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 103.

Claim 105

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the GPG program is a graphical program; wherein the GPG program includes at least one object creation node for automatically creating at least one graphical program object in the new graphical program; and wherein said generating the new graphical program comprises including the at least one graphical program object in the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:10, and col. 5:32-41, quoted above with regard to claim 69, as well as col. 7:22-36, col. 7:55-59, and Figure 37.

Nowhere does the cited text (or Ohara in general) teach or suggest that the program that generates the new program is a graphical program. In other words, Ohara does not disclose that the *generating* program (as opposed to the generated program) includes a plurality of interconnected nodes that visually represent the functionality of the program. Appellant respectfully submits that the Examiner has improperly characterized a program that utilizes graphics as a "graphical program". As is well known in the art of programming, both text-based and graphical programs may utilize graphics, e.g., graphical user interfaces, and so this is not a distinguishing characteristic of graphical programs in general. Thus, Appellant respectfully submits that the cited text fails to teach that the GPG program is a graphical program.

Nor do the cited passages disclose a GPG program that includes at least one object creation node for automatically creating at least one graphical program object in the new graphical program. In fact, nowhere does Ohara even mention an object creation node at all, and specifically fails to disclose one that is executable in the GPG to include the at least one graphical program object in the new graphical program. Nor does the cited Figure 37 disclose these features of claim 105.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 105.

Claim 106

In addition to the distinctions noted above with regard to claims 105 and 65, the cited art fails to teach "wherein the GPG program further includes a property node, the

method further comprising: the property node getting or setting a property of the graphical program object in response to said executing the GPG program".

In asserting that Ohara teaches these features, the Examiner cites col. 17:22-36, and col. 17:55-59; and Figure 37. However, the cited text actually reads:

> "On the other hand, the `SET/REST` behavioral state applies for example to an imagined case in which operations of the selected output signal is used to drive the second needle of a stop watch.
>
> Subsequently, the user enters a comment 303 to a character input area, a field commonly known in the visual interface environment. In the case of creation of a program implementing a quiz response system, for example, the comment can be used for giving a name of `correct answer lamp` to the output signal Y000. The user enters a string of characters as a comment 303 to the character input area via the keyboard 22. The comment 303 is not absolutely required though. That is to say, a comment 303 does not have to be entered. In the above description, the behavioral characteristic 301, the behavioral state 302 and the comment 303 are entered in an order they are enumerated here."

and

> ...a parameter setting window 400 used for setting parameters of the behavioral characteristics 301 selected by using the behavioral characteristic selection window 300 shown in FIG. 4. On the parameter setting window 400, a `delay` parameter 410 corresponds to the `A delay exists` item of the behavioral characteristics 301 of the behavioral characteristic selection window 300 shown in FIG. 4 whereas a `flickering` parameter 420 corresponds to the `flickering` item of the behavioral characteristics 301 of the behavioral characteristic selection window 300.

As may be seen, the cited text in no way discloses a property node, nor including such a property node in a GPG program, nor executing the GPG program to get or set a property of the graphical program object via the property node. Nor does Figure 37 disclose these features of claim 106.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 106.

Claim 107

In addition to the distinctions noted above with regard to claims 106, 105, and 65, the cited art fails to teach "wherein the object creation node outputs a reference to the graphical program object; wherein the property node receives the reference as input to the graphical program object; and wherein the property node gets or sets a property of the graphical program object specified by the reference to the graphical program object".

In asserting that Ohara teaches these features, the Examiner cites col. 17:22-36, and col. 17:55-59; and Figure 37. However, the cited text, quoted above with regard to claim 106, in no way discloses a property node at all, nor an object creation node (included in the GPG program) creating, and outputting a reference to, a graphical program object, nor the property node setting or getting a property of the graphical program object specified by the reference. Nor does Figure 37 disclose these features of claim 107.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 107.

Claim 108

In addition to the distinctions noted above with regard to claims 105 and 65, the cited art fails to teach "wherein the GPG program further includes an invoke node; the method further comprising: the invoke node invoking a method on the graphical program object in response to said executing the GPG program".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8 37. However, the cited text actually reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

Clearly, the cited text does not teach or suggest, or even hint at, a GPG program including an invoke node, nor the invoke node invoking a method on the graphical program object in response to said executing the GPG program, as claimed.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 108.

Claim 109

In addition to the distinctions noted above with regard to claims 108, 105, and 65, the cited art fails to teach "wherein the object creation node outputs a reference to the graphical program object; wherein the invoke node receives as input the reference to the graphical program object; and wherein the invoke node invokes a method on the graphical program object specified by the reference to the graphical program object".

In asserting that Ohara teaches these features, the Examiner cites col. 25:5-8 37. However, the cited text actually reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

Clearly, the cited text does not teach or suggest, or even hint at, a GPG program including an invoke node at all, nor an object creation node (included in the GPG program) creating, and outputting a reference to, a graphical program object, nor the invoke node invoking a method on the graphical program object specified by the reference.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 109.

Claim 110

In addition to the distinctions noted above with regard to claims 105 and 65, the cited art fails to teach "configuring the object creation node of the GPG program; wherein said configuring comprises specifying a graphical program object class for the object creation node; and wherein the at least one graphical program object included in the new graphical program is of the graphical program object class".

In asserting that Ohara teaches these features, the Examiner cites col. 17:22-36, and col. 17:55-59; and Figure 37. However, as noted above, the cited text actually reads:

> "On the other hand, the `SET/REST` behavioral state applies for example to an imagined case in which operations of the selected output signal is used to drive the second needle of a stop watch.
>
> Subsequently, the user enters a comment 303 to a character input area, a field commonly known in the visual interface environment.

> In the case of creation of a program implementing a quiz response system, for example, the comment can be used for giving a name of `correct answer lamp` to the output signal Y000. The user enters a string of characters as a comment 303 to the character input area via the keyboard 22. The comment 303 is not absolutely required though. That is to say, a comment 303 does not have to be entered. In the above description, the behavioral characteristic 301, the behavioral state 302 and the comment 303 are entered in an order they are enumerated here."

and

> ...a parameter setting window 400 used for setting parameters of the behavioral characteristics 301 selected by using the behavioral characteristic selection window 300 shown in FIG. 4. On the parameter setting window 400, a `delay` parameter 410 corresponds to the `A delay exists` item of the behavioral characteristics 301 of the behavioral characteristic selection window 300 shown in FIG. 4 whereas a `flickering` parameter 420 corresponds to the `flickering` item of the behavioral characteristics 301 of the behavioral characteristic selection window 300.

As may be seen, the cited text in no way discloses an object creation node of a GPG program at all, and more specifically fails to disclose configuring the object creation node of the GPG program, including specifying a graphical program object class for the object creation node, where the at least one graphical program object included in the new graphical program is of the graphical program object class. Nor does Figure 37 disclose these features of claim 110.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 110.

Claim 111

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the GPG program is a graphical program; and wherein the GPG program includes a graphical program creation node for automatically creating the new graphical program".

In asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:10, col. 5:32-41, col. 7:22-36, col. 7:55-59, quoted above with regard to claim 71 and 105, and Figure 37.

Nowhere does the cited text (or Ohara in general) teach or suggest that the program that generates the new program is a graphical program. In other words, Ohara does not disclose that the *generating* program (as opposed to the generated program) includes a plurality of interconnected nodes that visually represent the functionality of the program. Appellant respectfully submits that the Examiner has improperly characterized a program that utilizes graphics as a "graphical program". As is well known in the art of programming, both text-based and graphical programs may utilize graphics, e.g., graphical user interfaces, and so this is not a distinguishing characteristic of graphical programs in general. Thus, Appellant respectfully submits that the cited text fails to teach that the GPG program is a graphical program.

Nor do the cited passages disclose a GPG program that includes a graphical program creation node for automatically creating the new graphical program. In fact, nowhere does Ohara even mention a graphical program creation node at all, and specifically fails to disclose one that is executable in the GPG to create or instantiate the new graphical program. Nor does the cited Figure 37 disclose these features of claim 111.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 111.

Claim 112

Claim 112 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

The Examiner asserts that Ohara discloses all the features and limitations of claim 112, and further asserts that claims 112-115 are simply further computer-implement method versions of claim 65, 66, 77, and 81, respectively. Appellant respectfully disagrees, noting that claim 65 is directed to a method for automatically generating a new graphical program, whereas claim 112 is directed to automatically *modifying* an existing graphical program, which is quite different. Appellant respectfully submits that there are numerous limitations of claim 112 not taught or suggested by Ohara.

For example, following the argumentation presented above with regard to claim 65, the Examiner asserts that Ohara teaches "the GPG program receiving information,

wherein the information specifies functionality to add to the existing graphical program", citing Figure 4 and Figure 6, but omits the claimed feature "wherein the information does not specify specific objects to add to the existing graphical program".

Appellant submits that while Figures 4 and 6 disclose user input specifying functionality of the program, these figures (and Ohara in general) fail to teach "wherein the information does not specify specific objects to add to the existing graphical program" as claimed. Figure 4 is described thusly (col. 16:66 – col. 17:10):

> FIG. 4 is an ***explanatory diagram showing a behavioral characteristic selection window 300 for selecting a behavior of the output signal Y000 selected from the signal box 200***. The behavioral characteristic selection window 300 shown in FIG. 4 is displayed on the display apparatus 1 when a signal set item of a set, menu on the signal box 200 shown in FIG. 3 is selected. It should be noted that the signal set item itself is not shown in FIG. 3.
>
> ***The user selects and enters a behavioral characteristic 301, a behavioral state 302 and a comment 303 of the object defining a behavior, which has been selected from the signal box 200*** shown in FIG. 3, as follows. (*emphasis added*)

Figure 6 is described thusly:

> FIG. 6 is an explanatory diagram showing a relevant signal selection window 500, a window used for selecting signals relevant to the behavior of the output signal. ***A signal relevant to the behavior of the output signal is selected from those shown in the signal box 200 described earlier by clicking a graphical object representing the signal by means of the mouse*** 21. ***A graphical object representing a selected signal is automatically displayed in a behavior relevant signal area*** 501 of the relevant signal selection window 500 as shown in FIG. 6. In the case of some contemporary visual programming tools, in such an operation to select an item signal, it is necessary to carry out an operation of dragging a graphical object representing the selected item to a predetermined position typically by means of a mouse. In the case of the visual programming method provided by the present invention and the system adopting the method, on the other hand; the dragging operation is not required.
>
> In the explanatory diagram of FIG. 6, ***input signals X000, X001 and X002 are selected by repeating the select operation described above***. The input signals X000, X001 and X002 are signals generated by on/off switches for operating the output signal Y000. The switches correspond to switches of a generally used illuminator.
> (*emphasis added*)

Nowhere do Figures 4 and 6, nor the related text (or Ohara in general) disclose these features of claim 112. Rather, the Figures and cited text emphasize the fact that direct user input selects virtually all aspects of the generated program, as is also clearly described elsewhere. For example, Figure 2 and related text describe the basic method of Ohara, and disclose user input specifying objects for the program:

> With the visual programming method provided by the present invention, in order to automatically generate a program of a PLC, ***the user executes the steps of selecting an output signal defining a behavior; selecting a behavior; selecting signals relevant to the behavior; setting behavioral rules; and confirming the behavior***. For operations carried out to execute the steps, the use of various input units can be thought of. Graphical objects such as a button, a check box, a slide bar and an icon are operated mainly by clicking or dragging the mouse 21 of the input apparatus 2. On the other hand, a string of characters is entered mainly by using the keyboard 22 of the input apparatus 2. (*emphasis added*)

More particularly disclosing such user selection, col. 6:19-23 reads:

> Further, a system adopting the visual programming method according to the present invention comprises:
>
> an object selection means which displays graphical objects each defining a behavior on a graphical editor and which allows ***a user to select those of the displayed graphical objects as objects to be used in the programming work***;
>
> a behavior selection means which displays behavioral characteristics of the graphical objects and which allows ***the user to select a behavioral characteristic of each of the selected graphical objects***;
>
> a relevant object selection means which defines a relation among the plurality of graphical objects selected by using the object selection means and the behavior selection means; and
>
> a behavioral rule setting means which sets a behavioral rule of the selected graphical objects.
> (*emphasis added*)

Additionally, claim 1 of Ohara recites in pertinent parts:

> 1. A visual programming method comprising:

a first step of displaying graphical objects representing a plurality of members connected to output terminals of a programmable logic controller;

a second step of displaying graphical objects representing the output terminals of the programmable logic controller;

a third step of ***connecting a first graphical object selected by a user from the graphical objects representing the plurality of members displayed in the first step, to a second graphical object selected by the user*** from the graphical objects representing the output terminals and displayed in the second step;

a fourth step of ***allowing a user to select a graphical object or a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram;

a sixth step of creating a layout diagram showing a layout of the graphical objects representing the plurality of members and displaying said layout diagram on a screen;

a seventh step of ***allowing the user to select one of said graphical objects*** each used for defining a function and transferring said selected graphical object to said layout diagram created and displayed at said sixth step;

an eighth step of ***allowing a user to select a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram created at said sixth step;

a ninth step of ***allowing a user to select a graphical object or a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram created at said sixth step as well as displaying the same plurality of selected graphical objects on said layout diagram;

a tenth step of identifying arrangement orders of ***graphical objects selected by a user*** and assigning a priority to each of said arrangement orders when a plurality of arrangement orders are identified;

an eleventh step of displaying the same plurality of identified arrangement orders sequentially one after another in a sequence determined by said priorities assigned thereto at said tenth step,

a twelfth step of ***allowing the user to select an arrangement order among the same plurality of identified arrangement orders*** displayed sequentially at said eleventh step;

a thirteenth of ***allowing a user to change an arrangement order of graphical objects selected by the user*** at said twelfth step;

a fourteenth step of detecting a settable parameter graphical object with a new parameter thereof ***among graphical objects selected by a user*** and notifying the user of a result of detection;

> a fifteenth step of detecting selection of said graphical object reported to the user at said fourteenth step or a graphical object other than said reported graphical object and creating and displaying a window used for setting a parameter of said reported graphical object or said other graphical object; and
>
> a sixteenth step of ***allowing the user to set said parameter*** on said window displayed at said fifteenth step.

Moreover, as Ohara states in col. 41:16-21:

> To be more specific, the user is capable of generating a program through user interfaces. In addition, the user is also capable of carrying out simulation of a generated program by carrying out the same operations as the programming. Furthermore, it is needless to say that ***the user is also capable of modifying a program***. (*emphasis added*)

In other words, Ohara does not disclose modifying an existing graphical program in the manner claimed, i.e., based on information that does not specify specific objects to add to the existing graphical program.

Thus, Appellant submits that Ohara does not teach this feature of claim 112.

The Examiner further asserts that Ohara teaches "wherein said automatically modifying the existing graphical program modifies the existing graphical program without direct user input specifying the modification to the existing graphical program during said modifying", citing col. 16:27-28, and col. 17:26-39. However, col. 16:27-28 actually reads:

> "In the following description, only matters related to the user interface characterizing the visual programming method are explained, excluding automatic generation of a program ***because the automatic generation of a program is the same as the conventional system***." (*emphasis added*)

In other words, Ohara does not consider the automatic generation of the program in Ohara's system and method to be novel, and does not disclose how it is performed, other than to say it is performed according to prior art approaches.

Col. 17:26-39 reads:

> Subsequently, the user enters a comment 303 to a character input area, a field commonly known in the visual interface environment. In the case of creation of a program implementing a quiz response system, for example, the comment can be used for giving a name

> of `correct answer lamp` to the output signal Y000. The user enters a string of characters as a comment 303 to the character input area via the keyboard 22. The comment 303 is not absolutely required though. That is to say, a comment 303 does not have to be entered. In the above description, the behavioral characteristic 301, the behavioral state 302 and the comment 303 are entered in an order they are enumerated here. It should be noted, however, that they can be entered in an order freely selected by the user.

Clearly, the cited text in no way discloses "wherein said automatically modifying the existing graphical program modifies the existing graphical program without direct user input specifying the modification to the existing graphical program during said modifying", as recited in claim 112.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 112. Appellant notes that the above arguments are also applicable to the other independent claims grouped therewith.

Claim 113

In addition to the distinctions noted above with regard to claim 112, the cited art fails to teach "wherein said automatically modifying the existing graphical program modifies the existing graphical program without direct user input specifying modification to the plurality of nodes and without direct user input specifying modification to the interconnections between the plurality of nodes".

Following the argumentation presented above with regard to claim 66, in asserting that Ohara teaches these features, the Examiner cites col. 16:27-28, and col. 17:26-39. However, as shown above with regard to claim 112, the cited text in no way discloses this feature. Nor, as argued at length above, does Ohara anywhere teach or suggest this feature, but rather specifically describes user-selection of objects in the generated program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 113.

Claim 114

In addition to the distinctions noted above with regard to claim 112, the cited art fails to teach "wherein said modifying the existing graphical program comprises adding graphical code to the existing graphical program".

Following the argumentation presented above with regard to claim 77, in asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claims 65 and 112. However, the cited text does not teach or suggest, or even hint at, automatically modifying an existing graphical program by adding graphical code to the existing graphical program in the manner claimed, i.e., based on information that does not specify specific objects to add to the existing graphical program. Rather, as argued at length above, in Ohara's system, the user selects the objects to add to the program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 114.

Claim 115

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein said receiving information during program execution comprises receiving user input specifying desired functionality to add to the existing graphical program".

Following the argumentation presented above with regard to claim 81, in asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claims 112 and 65. However, the cited text does not teach or suggest, or even hint at, receiving *user input specifying desired functionality* to *add* to an existing graphical program. In fact, nowhere does Ohara mention or even hint at automatically modifying an existing graphical program in the manner claimed, i.e., based on user input that specifies desired functionality to add to the existing graphical program, and that does not specify specific objects to add to the existing graphical program. Rather, as argued at length above, in Ohara's system, the user selects the objects to add to the program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 115.

Claims 118, 135, 140, 142

Claim 118 is separately patentable because the cited reference does not teach or suggest the limitations recited in this claim.

Following the arguments presented above with regard to claim 65, the Examiner asserts that Ohara teaches the limitation "receive information, wherein the information specifies functionality of the new graphical program", citing Figure 4 and Figure 6, but omits the claimed feature "wherein the information does not specify specific objects for the new graphical program".

Appellant submits that while Figures 4 and 6 disclose user input specifying functionality of the program, these figures (and Ohara in general) fail to teach "wherein the information does not specify specific objects for the new graphical program" as claimed. Figure 4 is described thusly (col. 16:66 – col. 17:10):

> FIG. 4 is an ***explanatory diagram showing a behavioral characteristic selection window 300 for selecting a behavior of the output signal Y000 selected from the signal box 200***. The behavioral characteristic selection window 300 shown in FIG. 4 is displayed on the display apparatus 1 when a signal set item of a set, menu on the signal box 200 shown in FIG. 3 is selected. It should be noted that the signal set item itself is not shown in FIG. 3.
>
> ***The user selects and enters a behavioral characteristic 301, a behavioral state 302 and a comment 303 of the object defining a behavior, which has been selected from the signal box 200*** shown in FIG. 3, as follows. (*emphasis added*)

Figure 6 is described thusly:

> FIG. 6 is an explanatory diagram showing a relevant signal selection window 500, a window used for selecting signals relevant to the behavior of the output signal. ***A signal relevant to the behavior of the output signal is selected from those shown in the signal box 200 described earlier by clicking a graphical object representing the signal by means of the mouse*** 21. ***A graphical object representing a selected signal is automatically displayed in a behavior relevant signal area*** 501 of the relevant signal selection window 500 as shown in FIG. 6. In the case of some contemporary visual programming tools, in such an operation to select an item signal, it is necessary to carry out an operation of dragging a graphical object representing the selected item to a predetermined position typically by means of a mouse. In the case of the visual programming method provided by the present invention and the system adopting the

method, on the other hand; the dragging operation is not required.

In the explanatory diagram of FIG. 6, ***input signals X000, X001 and X002 are selected by repeating the select operation described above***. The input signals X000, X001 and X002 are signals generated by on/off switches for operating the output signal Y000. The switches correspond to switches of a generally used illuminator.
(*emphasis added*)

Nowhere do Figures 4 and 6, nor the related text (or Ohara in general) disclose these features of claim 118. Rather, the Figures and cited text emphasize the fact that direct user input selects virtually all aspects of the generated program, as is also clearly described elsewhere.

For example, Figure 2 and related text describe the basic method of Ohara, and disclose user input specifying objects for the program:

> With the visual programming method provided by the present invention, in order to automatically generate a program of a PLC, ***the user executes the steps of selecting an output signal defining a behavior; selecting a behavior; selecting signals relevant to the behavior; setting behavioral rules; and confirming the behavior***. For operations carried out to execute the steps, the use of various input units can be thought of. Graphical objects such as a button, a check box, a slide bar and an icon are operated mainly by clicking or dragging the mouse 21 of the input apparatus 2. On the other hand, a string of characters is entered mainly by using the keyboard 22 of the input apparatus 2. (*emphasis added*)

More particularly disclosing such user selection, col. 6:19-23 reads:

> Further, a system adopting the visual programming method according to the present invention comprises:
>
> an object selection means which displays graphical objects each defining a behavior on a graphical editor and which allows ***a user to select those of the displayed graphical objects as objects to be used in the programming work***;
>
> a behavior selection means which displays behavioral characteristics of the graphical objects and which allows ***the user to select a behavioral characteristic of each of the selected graphical objects***;
>
> a relevant object selection means which defines a relation among the

plurality of graphical objects selected by using the object selection means and the behavior selection means; and

a behavioral rule setting means which sets a behavioral rule of the selected graphical objects.
(*emphasis added*)

Additionally, claim 1 of Ohara recites in pertinent parts:

1. A visual programming method comprising:

a first step of displaying graphical objects representing a plurality of members connected to output terminals of a programmable logic controller;

a second step of displaying graphical objects representing the output terminals of the programmable logic controller;

a third step of ***connecting a first graphical object selected by a user from the graphical objects representing the plurality of members displayed in the first step, to a second graphical object selected by the user*** from the graphical objects representing the output terminals and displayed in the second step;

a fourth step of ***allowing a user to select a graphical object or a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram;

a sixth step of creating a layout diagram showing a layout of the graphical objects representing the plurality of members and displaying said layout diagram on a screen;

a seventh step of ***allowing the user to select one of said graphical objects*** each used for defining a function and transferring said selected graphical object to said layout diagram created and displayed at said sixth step;

an eighth step of ***allowing a user to select a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram created at said sixth step;

a ninth step of ***allowing a user to select a graphical object or a plurality of graphical objects from said graphical objects*** each used for defining a function and displayed at said first step and transferring the same plurality of selected graphical objects to said layout diagram created at said sixth step as well as displaying the same plurality of selected graphical objects on said layout diagram;

a tenth step of identifying arrangement orders of ***graphical objects selected by a user*** and assigning a priority to each of said arrangement orders when a plurality of arrangement orders are identified;

an eleventh step of displaying the same plurality of identified arrangement orders sequentially one after another in a sequence determined by said priorities assigned thereto at said tenth step,

a twelfth step of ***allowing the user to select an arrangement order among the same plurality of identified arrangement orders*** displayed sequentially at said eleventh step;

a thirteenth of ***allowing a user to change an arrangement order of graphical objects selected by the user*** at said twelfth step;

a fourteenth step of detecting a settable parameter graphical object with a new parameter thereof ***among graphical objects selected by a user*** and notifying the user of a result of detection;

a fifteenth step of detecting selection of said graphical object reported to the user at said fourteenth step or a graphical object other than said reported graphical object and creating and displaying a window used for setting a parameter of said reported graphical object or said other graphical object; and

a sixteenth step of ***allowing the user to set said parameter*** on said window displayed at said fifteenth step.

Thus, Appellant submits that Ohara does not teach this feature of claim 118.

The Examiner further asserts that Ohara teaches "wherein said automatically generating the new graphical program is performed without direct user input specifying the new graphical program", citing col. 16:27-28, and col. 17:26-39. However, col. 16:27-28 actually reads:

> "In the following description, only matters related to the user interface characterizing the visual programming method are explained, excluding automatic generation of a program ***because the automatic generation of a program is the same as the conventional system***." (*emphasis added*)

In other words, Ohara does not consider the automatic generation of the program in Ohara's system and method to be novel, and does not disclose how it is performed, other than to say it is performed according to prior art approaches.

Col. 17:26-39 reads:

> Subsequently, the user enters a comment 303 to a character input area, a field commonly known in the visual interface environment. In the case of creation of a program implementing a quiz response system, for example, the comment can be used for giving a name of `correct answer lamp` to the output signal Y000. The user enters

> a string of characters as a comment 303 to the character input area via the keyboard 22. The comment 303 is not absolutely required though. That is to say, a comment 303 does not have to be entered. In the above description, the behavioral characteristic 301, the behavioral state 302 and the comment 303 are entered in an order they are enumerated here. It should be noted, however, that they can be entered in an order freely selected by the user.

Clearly, the cited text in no way discloses "wherein said automatically generating the new graphical program is performed without direct user input specifying the new graphical program", as recited in claim 118.

Thus, for at least the reasons provided above, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 118. Appellant notes that the above arguments are also applicable to the other independent claims grouped therewith.

Claim 119

In addition to the distinctions noted above with regard to claim 119, the cited art fails to teach "wherein said automatically generating the new graphical program creates the new graphical program without direct user input selecting the plurality of nodes and without direct user input specifying the interconnections between the nodes".

Following the arguments presented above with regard to claim 66, in asserting that Ohara teaches these features, the Examiner cites col. 16:27-28, and col. 17:26-39. However, as shown above with regard to claim 65, the cited text in no way discloses this feature. Nor, as argued at length above, does Ohara anywhere teach or suggest this feature, but rather specifically describes user-selection of objects in the generated program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 66.

Claim 120

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach "wherein the new graphical program comprises a block diagram portion comprising the plurality of interconnected nodes and a user interface portion; wherein

said automatically generating the new graphical program includes generating the block diagram portion and the user interface portion".

Following the arguments presented above with regard to claim 67, in asserting that Ohara teaches these features, the Examiner cites Figure 17 and col. 3:46-56. However, Appellant notes that Figure 17 is described thusly:

> FIG. 17 is an explanatory diagram showing a portion of the component model window 160. On the component model window 160 shown in FIG. 17, typically, combinations of behavior relevant signals and behavioral characteristics displayed on the behavior verification window 1300 shown in FIG. 14 are expressed globally on a macro basis. Since the purpose of the behavior verification window 1300 is to locally verify operations carried out by a program generated for an output signal, a logic equation and behavioral characteristics are displayed in detail. On the other hand, the component model window 160 is used for verification of setting of behavioral characteristics for all output signals globally on a macro basis. Pieces of information set for behavioral characteristics of all output signals are thus displayed on the component model window 160 globally on a macro basis in order to allow all the pieces of information to be displayed at the same time. It should be noted that, in the example shown in FIG. 17, information on only one output signal is displayed though. (col. 38:65 – col. 39:16)

Appellant respectfully submits that Ohara nowhere describes Figure 17 as a graphical program including a block diagram portion comprising the plurality of interconnected nodes and a user interface portion. Rather, as described in the cited text, "the component model window 160 is used for verification of setting of behavioral characteristics for all output signals globally on a macro basis", where "Pieces of information set for behavioral characteristics of all output signals are thus displayed on the component model window 160 globally on a macro basis in order to allow all the pieces of information to be displayed at the same time." Appellant thus respectfully submits that the Examiner's characterization of Figure 17 as a graphical program with a block diagram and user interface portion is incorrect, and is not supported by the cited text.

Col. 3:46-56 reads:

> A function block diagram (FBD) is a program created by a language whereby function blocks (FBs), graphical objects each representing a function, are connected to each other to form a flow of data processing. A graphical editor for this language is a tool providing a work environment

> in which programming can be carried out by operating graphical objects displayed on a screen. The graphical objects include icons and characters used in the graphical user interface (GUI), a commonly known user interface for personal computers and workstations.

This text describes prior art function block diagrams that may be created by a user via a graphical editor, and in no way discloses an automatically generated graphical program that includes a block diagram and a user interface portion. The only user interface mentioned in this text is that of the work environment, i.e., of the programming environment.

Applicant respectfully submits that the Examiner has generally mischaracterized Ohara. For example, Applicant notes that the user interfaces described by Ohara here and elsewhere are not part of the generated program, but rather are comprised in various portions of Ohara's development system or tools used to generate and/or verify the generated program, e.g., Ohara's "behavior verification unit", mentioned in the cited portion of Ohara, Ohara's . As Ohara describes with reference to Figure 2 in col. 20, lines 50-60:

> At the step ST6, results of the operations carried out at the steps ST1 to ST5, that is, results of the programming work, are verified. When the setting of behavioral rules is finished, a behavior verification window 1300 shown in FIG. 14 is displayed on the display apparatus 1. On the behavior verification window 1300, the 3 input signals X000, X001 and X002, that is, the behavior relevant signals of the output signal Y000, are displayed as a rudder diagram. That is to say, the behavior verification window 1300 which displays behavioral characteristics of the output signal appears on the display apparatus 1.

Applicant notes that step ST6 of Figure 2 is simply a verification step in the development process of Ohara, and respectfully submits that the user interface of the behavior verification unit is *not* a user interface portion of Ohara's generated program. Applicant notes that Ohara's Figure 66 illustrates "a typical user interface of the behavior verification unit" (col. 61, lines 9-10). Applicant submits that Ohara's behavior verification unit is specifically *not* a portion of a generated program, but rather is a tool used to verify the functionality of the generated program, and notes that other tools for

aiding in the generation of the program described in Ohara include: a "behavior characteristic selection unit", "behavioral object selection unit", "behavioral condition setting unit", and a "relevant object selection unit". Applicant further notes that one or more of these additional tools also includes a respective graphical user interface which is similarly *not* a portion of the generated program. Applicant can find no reference teaching or suggesting that the generated program includes a graphical user interface portion, nor any reference teaching or suggesting that automatically generating the program includes generating a graphical user interface portion.

Additionally, Appellant notes that a program generated for, transferred to, and executed on, a programmable logic controller (PLC) would not include such a graphical user interface portion, such PLCs lacking display functionality.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 120.

Claims 125, 136

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach "wherein said automatically generating the new graphical program comprises: creating a plurality of graphical program objects in the new graphical program; and interconnecting the plurality of graphical program objects in the new graphical program; wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program".

Following the arguments presented above with regard to claim 68, in asserting that Ohara teaches these features, the Examiner cites Figure 17 and col. 3:46-56. However, nowhere does Figure 17 disclose automatically creating a plurality of graphical program objects in the new graphical program; and interconnecting the plurality of graphical program objects in the new graphical program. Rather, as described in the cited text, "the component model window 160 is used for verification of setting of behavioral characteristics for all output signals globally on a macro basis", where "Pieces of information set for behavioral characteristics of all output signals are thus displayed on the component model window 160 globally on a macro basis in order to allow all the pieces of information to be displayed at the same time." Appellant thus respectfully

submits that the Examiner's characterization of Figure 17 resulting from the *automatic creation* of a plurality of graphical program objects is incorrect, and is not supported by the cited text. Rather, as argued at length above, in Ohara's system, the *user* selects the graphical program objects.

Moreover, as also shown above, col. 3:46-56 describes prior art function block diagrams that may be created by a user via a graphical editor, and in no way discloses automatically generating a new graphical program by creating a plurality of graphical program objects in the new graphical program and interconnecting them, where the interconnected plurality of graphical program objects form at least a portion of the new graphical program, in the manner claimed.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 125.

Claims 126, 137

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach the limitation "automatically create one or more user interface objects in the new graphical program, wherein the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program".

Following the arguments presented above with regard to claim 69, in asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:10, and col. 5:32-41. However, col. 16:66 – col. 17:10 actually reads:

> FIG. 4 is an ***explanatory diagram showing a behavioral characteristic selection window 300 for selecting a behavior of the output signal Y000 selected from the signal box 200***. The behavioral characteristic selection window 300 shown in FIG. 4 is displayed on the display apparatus 1 when a signal set item of a set, menu on the signal box 200 shown in FIG. 3 is selected. It should be noted that the signal set item itself is not shown in FIG. 3.
>
> ***The user selects and enters a behavioral characteristic 301, a behavioral state 302 and a comment 303 of the object defining a behavior, which has been selected from the signal box 200*** shown in FIG. 3, as follows. (*emphasis added*)

Col. 5:32-41 reads:

> As a result, even a user who has a lack of professional knowledge of programming is capable of developing a program with ease.
>
> In addition, in the visual programming method according to the present invention, the step of selecting a behavioral characteristic of a graphical object displayed on the graphical editor further includes a step of entering a detailed parameter of the behavioral characteristic. As a result, ***the user is capable of changing a set value of a behavioral characteristic and, hence, creating a program*** with ease.
>
> Moreover, in the visual method according to the present invention, the step of setting a behavioral rule for graphical objects each defining a behavior with respect to the graphical objects further includes a step of generating a behavioral rule not set yet on the graphical editor from already set behavioral rules. As a result, ***the user is capable of creating a program*** with ease without the need to take all behavioral rules into consideration. (*emphasis added*)

As the cited text (and the text and claim quoted above with regard to claim 118) clearly indicates, Ohara's system and method relies on direct user input specifying virtually all aspects of the generated program, and nowhere discloses automatic generation of a graphical in the manner claimed.

For example, nowhere does Ohara disclose that the generated program includes a user interface. As argued above, the only user interfaces disclosed by Ohara are those of the work environment, i.e., of the programming environment. In other words, Applicant respectfully submits that the Examiner has generally mischaracterized Ohara, in that the user interfaces described by Ohara are not part of the generated program, but rather are comprised in various portions of Ohara's development system or tools used to generate and/or verify the generated program. Applicant can find no reference teaching or suggesting that the generated program includes automatically generated user interface objects for providing input to and/or displaying output from the generated program.

Additionally, as noted above, a program generated for, transferred to, and executed on, a programmable logic controller (PLC) would not include such user interface objects, such PLCs lacking display functionality.

Thus, nowhere does the cited text (or Ohara in general) disclose automatically generating the new graphical program by creating one or more user interface objects in

the new graphical program, where the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 126.

Claims 127, 138

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach "wherein the new graphical program implements a measurement function".

Ohara nowhere teaches these features. In fact, Ohara never even mentions measurement or a measurement function at all, and so cannot teach this limitation of claim 127

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 79.

Claims 128, 139

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach the limitation "receive user input specifying desired functionality of the new graphical program, wherein the generated new graphical program implements the specified desired functionality".

Following the argumentation presented above with regard to claim 81, in asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claims 112 and 65. However, the cited text does not teach or suggest, or even hint at, receiving *user input specifying desired functionality* of the new graphical program in the manner claimed, i.e., based on user input that not specify specific objects for the new graphical program. Rather, as argued at length above, in Ohara's system, the user selects the objects to include in the generated program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 128.

Claim 130

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach "wherein the received information comprises information regarding an existing program having program functionality; and wherein the generated new graphical program implements at least a portion of the program functionality of the existing program".

Following the argumentation presented above with regard to claim 81, in asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claim 65. However, the cited text does not teach or suggest, or even hint at, receiving an existing program (with program functionality) specifying desired functionality of the new graphical program, nor generating a new graphical program that implements at least a portion of the program functionality of the existing program. In fact, nowhere does Ohara mention or even hint at automatically generating a new graphical program based on an existing program.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 130.

Claims 132, 143

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the generated new graphical program implements additional functionality in addition to the functionality specified by the received information".

Following the argumentation presented above with regard to claim 85, in asserting that Ohara teaches these features, the Examiner cites col. 25:5-8. However, the cited text actually reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

Clearly, the cited text does not teach or suggest, or even hint at, the generated new graphical program implementing additional functionality in addition to the functionality specified by the received information.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 132.

Claims 133, 144

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach "wherein the new graphical program implements only a portion of the specified functionality".

Following the argumentation presented above with regard to claim 93, in asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

The cited text does not teach or suggest, or even hint at, an automatically generated graphical program that implements only a portion of the specified functionality, as claimed.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 133.

Claims 134, 145

In addition to the distinctions noted above with regard to claim 118, the cited art fails to teach "wherein the new graphical program is a partial program, the method further comprising: adding additional graphical code to the new graphical program, in response to manual user input, in order to complete the new graphical program".

Following the argumentation presented above with regard to claim 94, in asserting that Ohara teaches these features, the Examiner cites col. 25:5-8, which reads:

> Like the conventional system, the first embodiment is of course provided with a means for generating a program or code even though this means is shown in none of the figures.

The cited text does not teach or suggest, or even hint at these claimed features.

Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 134.

Claim 141

In addition to the distinctions noted above with regard to claim 65, the cited art fails to teach "wherein the received information comprises information regarding an existing program having program functionality; and wherein the generated new graphical program implements at least a portion of the program functionality of the existing program".

Following the argumentation presented above with regard to claim 81, in asserting that Ohara teaches these features, the Examiner cites col. 16:66 – col. 17:6, quoted above with regard to claim 65. However, the cited text does not teach or suggest, or even hint at, receiving an existing program (with program functionality) specifying desired functionality of the new graphical program, nor generating a new graphical program that implements at least a portion of the program functionality of the existing program. In fact, nowhere does Ohara mention or even hint at automatically generating a new graphical program based on an existing program.

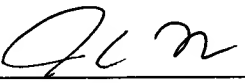
Thus, for at least these reasons, Appellant submits that Ohara fails to teach or suggest all the features and limitations of claim 141.

VIII. CONCLUSION

For the foregoing reasons, it is submitted that the Examiner's rejection of claims 65-145 was erroneous, and reversal of his decision is respectfully requested.

The Commissioner is authorized to charge the appeal brief fee of $500.00 and any other fees that may be due to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5150-44100/JCH. This Appeal Brief is submitted with a return receipt postcard.

Respectfully submitted,



Jeffrey C. Hood
Reg. No. 35,198
ATTORNEY FOR APPLICANT(S)

Meyertons Hood Kivlin Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8800

Date: 1/17/2006 JCH/MSW

IX. CLAIMS APPENDIX

The following lists claims 1-145 as incorporating entered amendments, including claims 65-145 as on appeal.

1.-64. (Cancelled).

65. (Previously Presented) A computer-implemented method for automatically generating a new graphical program, comprising:

executing a graphical program generation (GPG) program;

the GPG program receiving information, wherein the information specifies functionality of the new graphical program, wherein the information does not specify specific objects for the new graphical program; and

the GPG program automatically generating the new graphical program in response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality, and wherein the new graphical program comprises a plurality of interconnected nodes that visually indicate the functionality of the new graphical program;

wherein said automatically generating the new graphical program is performed without direct user input specifying the new graphical program.

66. (Previously Presented) The method of claim 65, wherein said automatically generating the new graphical program is performed without direct user input selecting the plurality of nodes and without direct user input specifying the interconnections between the plurality of nodes.

67. (Previously Presented) The method of claim 65, wherein the new graphical program comprises a block diagram portion comprising the plurality of interconnected nodes and a user interface portion; and

wherein said automatically generating the new graphical program includes automatically generating the block diagram portion and the user interface portion.

68. (Previously Presented) The method of claim 65, wherein said automatically generating the new graphical program comprises:

creating a plurality of graphical program objects in the new graphical program; and

interconnecting the plurality of graphical program objects in the new graphical program;

wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program.

69. (Previously Presented) The method of claim 65, wherein said automatically generating the new graphical program comprises:

creating one or more user interface objects in the new graphical program, wherein the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program.

70. (Previously Presented) The method of claim 65,

wherein the new graphical program is a virtual instrument.

71. (Previously Presented) The method of claim 65,

wherein the GPG program is a graphical program.

72. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program specifies a computational process; and

wherein the GPG program is operable to generate a new graphical program that implements the specified computational process.

73. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program specifies an algorithm; and

wherein the GPG program is operable to generate a new graphical program that implements the specified algorithm.

74. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program specifies a state diagram; and

wherein the GPG program is operable to generate a new graphical program that implements the specified state diagram.

75. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program specifies a prototype; and

wherein the GPG program is operable to generate a new graphical program that implements the specified prototype.

76. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program specifies a test executive sequence; and

wherein the GPG program is operable to generate a new graphical program that implements the specified test executive sequence.

77. (Previously Presented) The method of claim 65,

wherein said GPG program receiving information comprises the GPG program receiving user input specifying desired functionality of the new graphical program; and

wherein the GPG program is operable to generate a new graphical program that implements the specified desired functionality.

78. (Previously Presented) The method of claim 77,

wherein the GPG program comprises a graphical programming development environment application.

79. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program specifies an instrumentation function; and

wherein the GPG program is operable to generate a new graphical program that implements the specified instrumentation function.

80. (Previously Presented) The method of claim 79,

wherein the instrumentation function comprises one or more of:

a test and measurement function; or

an industrial automation function.

81. (Previously Presented) The method of claim 65,

wherein the information received by the GPG program comprises information regarding an existing program having program functionality; and

wherein the GPG program is operable to generate a new graphical program that implements at least a portion of the program functionality of the existing program.

82. (Previously Presented) The method of claim 81,

wherein the existing program is a graphical program.

83. (Previously Presented) The method of claim 65,

wherein the GPG program is operable to generate a plurality of new graphical programs, depending on the received information.

84. (Previously Presented) The method of claim 65,

wherein the new graphical program generated by the GPG program has program functionality; and

wherein the GPG program is operable to determine at least a portion of the program functionality independently of the received information.

85. (Previously Presented) The method of claim 65,

wherein the GPG program is operable to generate the new graphical program such that the new graphical program implements additional functionality in addition to the functionality specified by the received information.

86. (Previously Presented) The method of claim 65,

wherein the new graphical program comprises graphical program code; and

wherein the GPG program is operable to receive code generation information specifying how to generate at least a portion of the graphical program code.

87. (Previously Presented) The method of claim 65,

wherein said GPG program automatically generating the new graphical program comprises the GPG program calling an application programming interface (API) enabling the automatic generation of a graphical program.

88. (Previously Presented) The method of claim 65,

wherein said GPG program automatically generating the new graphical program comprises the GPG program automatically requesting a server program to generate the new graphical program.

89. (Previously Presented) The method of claim 88,

wherein the server program is an application instance of a graphical programming environment.

90. (Previously Presented) The method of claim 65,

wherein the GPG program comprises a client portion and a server portion; and

wherein the client portion is operable to utilize an application programming interface (API) in order to direct the server program to automatically generate the new graphical program.

91. (Previously Presented) The method of claim 90,

wherein the client portion of the GPG program executes in a first computer system;

wherein the server portion of the GPG program executes in a second computer system; and

wherein the first computer system is connected to the second computer system.

92. (Previously Presented) The method of claim 65, further comprising:

executing the new graphical program;

wherein the new graphical program is operable to perform the specified functionality during execution.

93. (Previously Presented) The method of claim 65,

wherein the new graphical program implements only a portion of the specified functionality.

94. (Previously Presented) The method of claim 65, wherein the new graphical program is a partial program, the method further comprising:

adding additional graphical code to the new graphical program, in response to manual user input, in order to complete the new graphical program.

95. (Previously Presented) The method of claim 65,

wherein said automatically generating the new graphical program comprises including at least one graphical program object in the new graphical program.

96. (Previously Presented) The method of claim 95, wherein the new graphical program includes a block diagram, wherein the at least one graphical program object comprises a function node placed in the block diagram.

97. (Previously Presented) The method of claim 95, wherein the new graphical program includes a block diagram, wherein the at least one graphical program object comprises a automatic structure placed in the block diagram.

98. (Previously Presented) The method of claim 95, wherein the new graphical program includes a user interface panel, wherein the at least one graphical program object comprises a user interface object placed in the user interface panel.

99. (Previously Presented) The method of claim 98, wherein the user interface object is a user interface input object placed in the user interface panel for performing one or more of: viewing input to the new graphical program; or providing input to the new graphical program.

100. (Previously Presented) The method of claim 98, wherein the user interface object is a user interface output object placed in the user interface panel for viewing output of the new graphical program.

101. (Previously Presented) The method of claim 98, wherein the new graphical program also includes a block diagram, wherein the user interface object is a user interface input object placed in the user interface panel for performing one or more of: viewing input to the block diagram; or providing input to the new graphical program.

102. (Previously Presented) The method of claim 98, wherein the new graphical program also includes a block diagram, wherein the user interface object is a user interface output object placed in the user interface panel for viewing output from the block diagram.

103. (Previously Presented) The method of claim 65, wherein said automatically generating the new graphical program comprises:

including a first graphical program object and a second graphical program object in the new graphical program; and

connecting the first graphical program object to the second graphical program object.

104. (Previously Presented) The method of claim 103, wherein said connecting the first graphical program object to the second graphical program object comprises connecting an input of the first graphical program object to an output of the second graphical program object.

105. (Previously Presented) The method of claim 65,

wherein the GPG program is a graphical program;

wherein the GPG program includes at least one object creation node for automatically creating at least one graphical program object in the new graphical program; and

wherein said generating the new graphical program comprises including the at least one graphical program object in the new graphical program.

106. (Previously Presented) The method of claim 105, wherein the GPG program further includes a property node, the method further comprising:

the property node getting or setting a property of the graphical program object in response to said executing the GPG program.

107. (Previously Presented) The method of claim 106, wherein the object creation node outputs a reference to the graphical program object;

wherein the property node receives the reference as input to the graphical program object; and

wherein the property node gets or sets a property of the graphical program object specified by the reference to the graphical program object.

108. (Previously Presented) The method of claim 105, wherein the GPG program further includes an invoke node; the method further comprising:

the invoke node invoking a method on the graphical program object in response to said executing the GPG program.

109. (Previously Presented) The method of claim 108, wherein the object creation node outputs a reference to the graphical program object;

wherein the invoke node receives as input the reference to the graphical program object; and

wherein the invoke node invokes a method on the graphical program object specified by the reference to the graphical program object.

110. (Previously Presented) The method of claim 105, further comprising:

configuring the object creation node of the GPG program;

wherein said configuring comprises specifying a graphical program object class for the object creation node; and

wherein the at least one graphical program object included in the new graphical program is of the graphical program object class.

111. (Previously Presented) The method of claim 65,

wherein the GPG program is a graphical program; and

wherein the GPG program includes a graphical program creation node for automatically creating the new graphical program.

112. (Previously Presented) A computer-implemented method for automatically modifying an existing graphical program, comprising:

executing a GPG program;

the GPG program receiving information during program execution, wherein the information specifies functionality to add to the existing graphical program, wherein the information does not specify specific objects to add to the existing graphical program;

the GPG program automatically modifying the existing graphical program in order to implement the specified functionality, in response to receiving the information, wherein the existing graphical program comprises a plurality of interconnected nodes that visually indicate the specified functionality of the existing graphical program;

wherein said automatically modifying the existing graphical program modifies the existing graphical program without direct user input specifying the modification to the existing graphical program during said modifying.

113. (Previously Presented) The method of claim 112, wherein said automatically modifying the existing graphical program modifies the existing graphical program without direct user input specifying modification to the plurality of nodes and without direct user input specifying modification to the interconnections between the plurality of nodes.

114. (Previously Presented) The method of claim 112,

wherein said modifying the existing graphical program comprises adding graphical code to the existing graphical program.

115. (Previously Presented) The method of claim 112,

wherein said receiving information during program execution comprises receiving user input specifying desired functionality to add to the existing graphical program.

116. (Previously Presented) A computer-implemented method for automatically generating a new graphical program, comprising:

providing information specifying functionality of the new graphical program, wherein the information does not specify specific objects for the new graphical program;

executing a graphical program generation (GPG) program;

the GPG program automatically generating the new graphical program using said information, wherein the new graphical program implements the specified functionality,

and wherein the new graphical program comprises a plurality of interconnected nodes that visually indicate the functionality of the new graphical program;

wherein said automatically generating the new graphical program creates the new graphical program without direct user input specifying the new graphical program.

117. (Previously Presented) The method of claim 116, wherein said automatically generating the new graphical program creates the new graphical program without direct user input selecting the plurality of nodes and without direct user input specifying the interconnections between the nodes.

118. (Previously Presented) A memory medium for automatically generating a new graphical program, the memory medium comprising program instructions executable to:

receive information, wherein the information specifies functionality of the new graphical program, wherein the information does not specify specific objects for the new graphical program;

automatically generate the new graphical program in response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality, wherein the new graphical program comprises a plurality of interconnected nodes that visually indicate the functionality of the new graphical program, and wherein said automatically generating the new graphical program creates the new graphical program without direct user input specifying the new graphical program.

119. (Previously Presented) The memory medium of claim 118, wherein said automatically generating the new graphical program creates the new graphical program without direct user input selecting the plurality of nodes and without direct user input specifying the interconnections between the nodes.

120. (Previously Presented) The memory medium of claim 118, wherein the new graphical program comprises a block diagram portion comprising the plurality of interconnected nodes and a user interface portion;

wherein said automatically generating the new graphical program includes generating the block diagram portion and the user interface portion.

121. (Previously Presented) A system for automatically generating a new graphical program, the system comprising:

a processor coupled to a memory, wherein the memory stores a graphical program generation (GPG) program;

wherein the processor is operable to execute the GPG program in order to:

receive information specifying functionality of the new graphical program, wherein the information does not specify specific objects for the new graphical program; and

automatically generate the new graphical program in response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality, and wherein the new graphical program comprises a plurality of interconnected nodes that visually indicate the functionality of the new graphical program, and wherein the new graphical program is automatically generated without direct user input specifying the new graphical program.

122. (Previously Presented) The system of claim 121, wherein said automatically generating the new graphical program creates the new graphical program without direct user input selecting the plurality of nodes and without direct user input specifying the interconnections between the nodes.

123. (Previously Presented) The system of claim 121, wherein the new graphical program comprises a block diagram portion comprising the plurality of interconnected nodes and a user interface portion;

wherein said automatically generating the new graphical program includes generating the block diagram portion and the user interface portion.

124. (Previously Presented) The system of claim 121, wherein said automatically generating the new graphical program comprises:

creating a plurality of graphical program objects in the new graphical program without direct user input specifying the plurality of graphical program objects; and

interconnecting the plurality of graphical program objects in the new graphical program without direct user input specifying the interconnections between the nodes;

wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program.

125. (Previously Presented) The memory medium of claim 118, wherein, in automatically generating the new graphical program, the program instructions are executable to: automatically create a plurality of graphical program objects in the new graphical program; and

automatically interconnect the plurality of graphical program objects in the new graphical program;

wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program.

126. (Previously Presented) The memory medium of claim 118, wherein, in automatically generating the new graphical program, the program instructions are executable to:

automatically create one or more user interface objects in the new graphical program, wherein the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program.

127. (Previously Presented) The memory medium of claim 118,

wherein the new graphical program implements a measurement function.

128. (Previously Presented) The memory medium of claim 118, wherein, in receiving information, the program instructions are executable to:

receive user input specifying desired functionality of the new graphical program, wherein the generated new graphical program implements the specified desired functionality.

129. (Previously Presented) The memory medium of claim 118,

wherein said receiving and said generating are performed by a graphical programming development environment application.

130. (Previously Presented) The memory medium of claim 118,

wherein the received information comprises information regarding an existing program having program functionality; and

wherein the generated new graphical program implements at least a portion of the program functionality of the existing program.

131. (Previously Presented) The memory medium of claim 118, wherein the program instructions are further executable to:

automatically generate a plurality of new graphical programs, depending on the received information.

132. (Previously Presented) The memory medium of claim 118,

wherein the generated new graphical program implements additional functionality in addition to the functionality specified by the received information.

133. (Previously Presented) The memory medium of claim 118,

wherein the new graphical program implements only a portion of the specified functionality.

134. (Previously Presented) The memory medium of claim 118, wherein the new graphical program is a partial program, and wherein the program instructions are further executable to:

add additional graphical code to the new graphical program in response to manual user input in order to complete the new graphical program.

135. (Previously Presented) A computer-implemented method for automatically generating a new graphical program, comprising:

receiving information, wherein the information specifies functionality of the new graphical program, wherein the information does not specify specific objects for the new graphical program; and

automatically generating the new graphical program in response to said information specifying the functionality of the new graphical program, wherein the new graphical program implements the specified functionality, and wherein the new graphical program comprises a plurality of interconnected nodes that visually indicate the functionality of the new graphical program;

wherein said automatically generating the new graphical program is performed without direct user input including the plurality of nodes or connecting the plurality of nodes.

136. (Previously Presented) The method of claim 135, wherein said automatically generating the new graphical program comprises:

automatically creating a plurality of graphical program objects in the new graphical program; and

automatically interconnecting the plurality of graphical program objects in the new graphical program;

wherein the interconnected plurality of graphical program objects comprise at least a portion of the new graphical program.

137. (Previously Presented) The method of claim 135, wherein said automatically generating the new graphical program comprises:

automatically creating one or more user interface objects in the new graphical program, wherein the one or more user interface objects perform one or more of providing input to or displaying output from the new graphical program.

138. (Previously Presented) The method of claim 135,

wherein the new graphical program implements a measurement function.

139. (Previously Presented) The method of claim 135,

wherein said receiving information comprises receiving user input specifying desired functionality of the new graphical program; and

wherein the generated new graphical program implements the specified desired functionality.

140. (Previously Presented) The method of claim 135,

wherein said receiving and said generating are performed by a graphical programming development environment application.

141. (Previously Presented) The method of claim 135,

wherein the received information comprises information regarding an existing program having program functionality; and

wherein the generated new graphical program implements at least a portion of the program functionality of the existing program.

142. (Previously Presented) The method of claim 135, further comprising:

automatically generating a plurality of new graphical programs, depending on the received information.

143. (Previously Presented) The method of claim 135,

wherein the generated new graphical program implements additional functionality in addition to the functionality specified by the received information.

144. (Previously Presented) The method of claim 135,

wherein the new graphical program implements only a portion of the specified functionality.

145. (Previously Presented) The method of claim 135, wherein the new graphical program is a partial program, the method further comprising:

adding additional graphical code to the new graphical program, in response to manual user input, in order to complete the new graphical program.

X. EVIDENCE APPENDIX

No evidence submitted under 37 CFR §§ 1.130, 1.131 or 1.132 or otherwise entered by the Examiner is relied upon in this appeal.

XI. RELATED PROCEEDINGS APPENDIX

There are no related proceedings.